

Санкт-Петербургский государственный университет

Кафедра системного программирования

Группа 22.Б09-мм

Адаптация приложений на DSP под новую инфраструктуру

Белянин Георгий Олегович

Отчёт по учебной практике
в форме «Производственное задание»

Научный руководитель:
старший преподаватель кафедры информационно-аналитических систем, К. К. Смирнов

Санкт-Петербург
2024

Оглавление

Введение	3
1. Постановка задачи	4
2. Обзор	5
3. Реализация	7
4. Оценка производительности решения	10
Заключение	12
Список литературы	13

Введение

В последнее время робототехника все чаще встречается в обучении школьников. Для преподавания детям её основ используют образовательные конструкторы, один из которых — конструктор ТРИК¹ [7]. В наборах ТРИК поставляется контроллер, позволяющий работать с различной периферией: моторами, датчиками, камерами. Для того чтобы сделать работу с камерой полезной, контроллер должен уметь обрабатывать изображение в реальном времени. Поскольку контроллер ТРИК представляет собой встраиваемую систему на базе ARM926EJ-S, производительности процессора не хватает, для обработки изображения используется специальный модуль — DSP (Digital Signal Processor), предназначенный для высокопроизводительной обработки больших массивов данных.

На момент начала курсовой работы контроллер работал под управлением операционной системы на базе ядра Linux 3.6. С течением времени появилась необходимость привести его ПО к актуальным версиям. Ядро Linux было обновлено до версии 4.14, но алгоритмы обработки изображения не были перенесены. Компания Texas Instruments, производящая DSP, переделала всю инфраструктуру для коммуникации процессоров: стала использовать новый модуль ядра `remoteproc`, закончила поддержку фреймворка для запуска вычислений `Codec Engine` — поэтому с обновлением кода возникли проблемы.

В работе приводится решение проблемы выполнения алгоритмов обработки изображения на DSP при помощи актуальных методов взаимодействия с удаленными процессорами.

¹<https://trikset.com/>

1 Постановка задачи

Целью работы является запуск существующих алгоритмов для DSP на новых версиях ядра Linux. Для её выполнения были поставлены следующие задачи:

1. Определить причины, по которым существующий код не может работать с новой версией ядра;
2. Найти альтернативы устаревшим решениям по взаимодействию с DSP;
3. Адаптировать существующие алгоритмы для работы с подходящими библиотеками.

2 Обзор

Контроллер ТРИК работает на процессоре OMAP-L138 C6-Integra ARM + DSP от Texas Instruments. Со стороны ARM происходит запуск, конфигурация алгоритма, отправка исходного и получение результирующего изображения. В качестве DSP используется TMS320C674x [6], он проводит сложные вычисления на изображении. Компания Texas Instruments так же предоставляет SDK² и несколько вспомогательных библиотек с оптимизированными функциями для обработки изображений: `vlib`³, `imglib`⁴.

ТРИК умеет работать с двумя типами камер: USB веб-камерами и специализированными модулями камер на базе ov7670. Для получения кадров используется V4L2 (Video4Linux) [1], который остается актуальным и на новых версиях ядра. Для вывода изображений на контроллере установлен дисплей разрешением 320 x 240. Отрисовка на нём доступна через `framebuffer` и вызовы ядра [2].

Алгоритмы для старого ядра⁵ написаны на комбинации языков C и C++, используют связку библиотек, предоставляемую компанией Texas Instruments для коммуникации с DSP: `syslink` и `cmemk`. Однако это решение, по указаниям производителя, устарело [4], и, на замену предыдущему подходу, предлагается использовать стандартный модуль ядра `remoteproc`.

Существующая схема работы имеет недостатки: программа графического интерфейса работает со множеством различных ссылок, а исполняемые файлы алгоритмов содержат почти одинаковый код, за исключением кода, выполняемого на DSP.

Реализованы шесть различных алгоритмов, запускающихся через различные исполняемые файлы:

- Датчик движения (motion sensor);

²<https://www.ti.com/tool/download/PROCESSOR-SDK-RTOS-OMAPL138/06.03.00.106>

³https://software-dl.ti.com/libs/vlib/latest/index_FDS.html

⁴https://software-dl.ti.com/sdoemb/sdoemb_public_sw/imglib/latest/index_FDS.html

⁵<https://github.com/trikset/trik-media-sensors-dsp>

- Датчик линии на основе границ (edge line sensor);
- Датчик линии (line sensor);
- Датчик объектов (object sensor);
- Алгоритм, преобразующий камеру в несколько сенсоров цвета;
- Энкодер JPEG изображений.

У каждого алгоритма есть конфигурируемые параметры, в большинстве случаев это целевой цвет в пространстве HSV, и выходные параметры: обработанное изображение и несколько целых чисел, с различным назначением. Например, алгоритм датчика движения конфигурируется допустимым диапазоном цветов объекта, дорисовывает окружность вокруг целевого объекта, возвращает радиус окружности и положение ее центра. Входное изображение имеет формат YUV422P, тот же формат, что и изображение с веб-камеры. Выходное — RGB565X — пространство, используемое для отрисовки на дисплее ТРИК.

В коде алгоритмов для старого ядра используется фреймворк от Texas Instruments — **Codec Engine**. Он отвечает за запуск алгоритмов обработки изображений на DSP. Однако, в последних версиях SDK для OMAP-L138, он был убран из списка поставляемых. Старые версии **Codec Engine** не могут работать с новой инфраструктурой remoteproc.

3 Реализация

В четвертой мажорной версии ядра Linux для ARM появилась абстракция `device-tree` [3], содержащая описание аппаратной конфигурации. Чтобы ядро успешно обнаружило DSP, в `device-tree` необходимо указать адрес в памяти, по которому будет загружаться прошивка [5].

SDK последней версии от Texas Instruments включает в себя библиотеку IPC (Inter-Process Communication) для сообщений между различными процессорами. Для ее применения в новой версии ядра необходимо включить несколько модулей ядра: `remoteproc`, `da8xx remoteproc`, `virtio rpmsg bus`, `rpmsg proto` [5], отвечающих за межпроцессорное взаимодействие.

После этих настроек на контроллере ТРИК успешно запускаются пример из библиотеки IPC, демонстрирующий успешную отправку сообщений между процессорами ARM и DSP.

Большинство сложностей, испытанных на подготовительном этапе, были связаны с системой сборки от Texas Instruments. Это связано со сложностью линковки и множеством необходимых библиотек для компиляции программы на DSP. Поэтому для собственного проекта за основу была взята конфигурация из примеров.

Вместо устаревшей инфраструктуры предоставляемой `Codec Engine` была выбрана следующая последовательность выполнения алгоритма на DSP:

- Получить на стороне ARM изображение с веб-камеры/`ov7670` и скопировать его в общую область памяти;
- Передать при помощи библиотеки IPC сигнал о необходимости обработать изображение заданным алгоритмом;
- Запустить заданный алгоритм на DSP;
- Передать обработанное изображение и результаты обратно на сторону ARM;

- Скопировать полученное изображение в буфер экрана;
- Повторить шаги, до тех пор, пока не получен сигнал о завершении.

Команды, посылаемые между процессорами в синхронном режиме, отражены в UML диаграмме на рис 1.

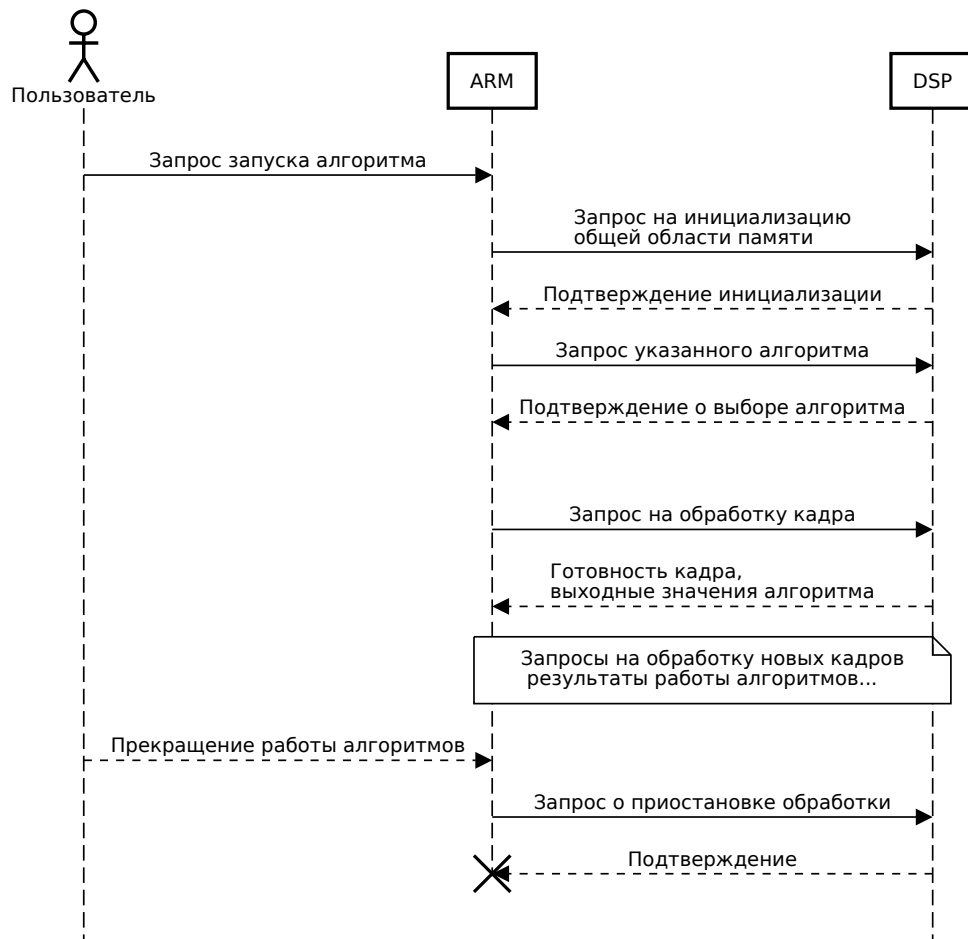


Рис. 1: UML диаграмма взаимодействия Пользователя/DSP/ARM

Из документации на процессор [6] видно, что DSP потенциально имеет прямой доступ к устройствам ввода/вывода и к оперативной памяти, следовательно, два полных копирования изображения кажутся излишними. Но DSP находится под управлением собственной небольшой операционной системы и не имеет драйверов для взаимодействия с framebuffer дисплея и различными камерами. Поэтому задача избавиться от двух лишних копирований нетривиальна.

Вместо прошлого решения, когда каждый алгоритм был отдельным исполняемым файлом, был выбран подход, в котором выбор алгорит-

ма осуществляется параметром командной строки. Для конфигурации используется внешний файл.

Существующие алгоритмы содержали в себе много дублирующегося кода, что было причиной некорректной работы алгоритмов `motion sensor`, `line sensor` и `mxn sensor`. В коде алгоритмов для нового ядра проведён рефакторинг, исправивший ошибки связанные с некорректными преобразованиями из пространства `YUV422P` в `RGB565X`.

4 Оценка производительности решения

Первые тесты были проведены на алгоритме датчика движения. Время обработки кадра стало заметно больше, чем на старом ядре. Отладка показала, что узким местом является выполнение расчётов на DSP. Причиной стали не оптимально выбранные флаги оптимизации. В связи с несовершенствами компилятора Texas Instruments для DSP на VLIW архитектуре часть из флагов работает контринтуитивно: `-O3` вместо `-O2` может привести DSP программу к зависанию, расстановка директив `#pragma` для сворачивания циклов может привести к ухудшению производительности. Из-за этой неопределенности флаги оптимизации были взяты из предыдущего проекта, директивы для оптимизаций составлялись экспериментально параллельно изучению сгенерированного ассемблерного листинга.

Разница во времени уменьшилась, но все равно визуально был заметен регресс производительности. Сравнение времени затрачиваемого на обработку кадра алгоритмами на старом и новом ядре приведено в таблице 1.

Название алгоритма	Время обработки кадра на новой инфраструктуре	Время обработки кадра на старой инфраструктуре
Object Sensor	273 мс/кадр	88 мс/кадр
Motion Sensor	121 мс/кадр	—
Edge-Line Sensor	113 мс/кадр	72 мс/кадр
Line Sensor	112 мс/кадр	80 мс/кадр

Таблица 1: Сравнение времени обработки кадра разными реализациями
“—” означает, что алгоритм не работал должным образом
стандартное отклонение составляет менее 1%

Измерения проводились при помощи аппаратных счетчиков на новой инфраструктуре и с использованием утилиты `strace` на старом ядре. Изучался набор из 16 последовательных кадров. Такой небольшой выборки достаточно для анализа причин ухудшения производительности, поскольку ввиду отсутствия условных конструкций в алгоритмах время обработки кадра не зависит от его содержания и стандартное

отклонение составляет менее 1%.

Новое время работы алгоритмов, по мнению заказчика, удовлетворительно для работы на уровне образовательной робототехники.

Впоследствии отладки с использованием встроенного счетчика тактов было выявлено, что запись результирующего изображения занимает большую часть времени. Предположительно, её можно сильно ускорить. Скорее всего, сейчас DSP использует strict memory ordering или его аналог, более медленный, чем relaxed memory ordering. Однако в силу недостаточной документации найти необходимые флаги не вышло.

Заключение

По итогам данной работы:

- Определены устаревшие зависимости `cmemk`, `syslink` и фреймворк `Codec Engine`
- Найдены актуальные альтернативы устаревшим решениям в виде библиотек IPC от Texas Instruments и модуля ядра `remoteproc`
- Все существующие алгоритмы компьютерного зрения перенесены в новый фреймворк и работают с достаточной производительностью.

Созданная инфраструктура в будущем может быть использована для запуска новых алгоритмов компьютерного зрения, работы с нейронными сетями и обработкой звука на DSP. Существующие алгоритмы можно сильно ускорить, опираясь на узкие места, указанные в работе.

Обновленная конфигурация параметров сборки ядра была интегрирована в систему сборки `open embedded`, соответствующие патчи были оформлены в виде `pull request` ⁶ ⁷.

Исходный код доступен на GitHub: <https://github.com/S007sayer/trik-media-sensors>

⁶<https://github.com/trikset/ti-linux-kernel/pull/14>

⁷<https://github.com/trikset/meta-trik/pull/154>

Список литературы

- [1] The Linux Kernel's documentation. — Дата обращения: 27.09.2023.
URL: https://www.kernel.org/doc/html/v4.9/media/media_kapi.html.
- [2] Linux Programmer's Manual. — Дата обращения: 28.09.2023.
- [3] Linux and Devicetree. — Дата обращения: 08.09.2023. URL: <https://docs.kernel.org/devicetree/usage-model.html>.
- [4] Processor SDK RTOS J721 documentation. — Дата обращения: 14.09.2023. URL: https://software-dl.ti.com/jacinto7/esd/processor-sdk-rtos-jacinto7/08_02_00_05/exports/docs/psdk_rtos/docs/user_guide/release_notes_08_02_00_j721e.html.
- [5] Texas Instruments. IPC Install Guide Linux. — Дата обращения: 08.09.2023. URL: <https://www.ti.com/tool/download/PROCESSOR-SDK-RTOS-OMAPL138/06.03.00.106>.
- [6] Texas Instruments. OMAP-L138 C6000 DSP+ARM Processor datasheet. — Дата обращения: 19.10.2023. URL: <https://www.ti.com/lit/gpn/OMAP-L138>.
- [7] Терехов Андрей Николаевич, Киселев Михаил Михайлович. Робототехнический конструктор трик на уроках “Технология” в школе // Современные информационные технологии и ИТ-образование. — 2015. — Vol. 1, no. 11. — P. 296–301. — Дата обращения: 11.12.2023.